UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/745,023	12/20/2000	Ram Kudukoli	5150-44100	8133

35690 7590 06/03/2004

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C.
P.O. BOX 398
AUSTIN, TX 78767-0398

EXAMINER	
KANG, INSUN	

ART UNIT	PAPER NUMBER
2124	7

DATE MAILED: 06/03/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/745,023	KUDUKOLI ET AL.	
	Examiner	Art Unit	
	Insun Kang	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 March 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-64* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-64* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☒ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *10/20/2003*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responding to amendment filed 3/24/2004.

2. As per applicant's request, claims 1, 44, 49, 53, 56 and 60 have been amended. Claims 1-64 are pending.

Oath/Declaration

3. A new oath or declaration has not been submitted. Therefore, a requirement of a new oath submission is maintained.

A new oath or declaration is required because the provisional application number is incorrect. The correct provisional application number should be 60/149,942. The wording of an oath or declaration cannot be amended. If the wording is not correct or if all of the required affirmations have not been made or if it has not been properly subscribed to, a new oath or declaration is required. The new oath or declaration must properly identify the application of which it is to form a part, preferably by application number and filing date in the body of the oath or declaration. See MPEP §§ 602.01 and 602.02.

Double Patenting

4. The applicant fails to show the reasons to traverse the double patenting rejection. Therefore, the rejection of double patenting is maintained.

Claims 1-64 are provisionally rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over copending applications U.S. Patent No. 09/595003 and 09/518,492. Although the conflicting claims are not

identical, they are not patentably distinct from each other because: application 09/595003 claims 1-6, 8-15, 17-23, 29-40, 42, and 47-64 generating a graphical program (09/595003; claim 1), receiving information (claim 2), manual user input, block diagram (claim 11), algorithm, prototype (claim 4, 6), user interface panel (claim 11), etc. Application 09/518,492 claims 1-8 and 14-64 generating a graphical program without user input ('492; claim 67, 68, 138, 139, 160, 165, 166), virtual instrument (claims 71, 142), user interface portion (claims 70, 141), block diagram (claims 70 and 141), displaying output (claims 74,144), invoking node (claims 113, 114), modifying the existing graphical program (claims 115, 137), property node (claims 126-129), API (claims 151, 155), etc.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-64 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kodosky et al. (US 5,732,277) in view of Uczekaj et al. (US 5,920,718).

Regarding claim 1, Kodosky et al. disclose: programmatically generating a new graphical program ("computer-generated display of a completed block diagram ...This block diagram is the graphical program representing the instrument's operation," col 16, lines 61-67) comprising: executing a graphical program generation (GPG) program ("advantageously permits the implementation of a system in which execution instructions can be constructed in a graphical fashion.... the execution instructions can be constructed in response to the construction of a block diagram comprising the graphical display (col 13, lines 40-62)"; The execution subunit executes execution instructions which are constructed in response to the graphical images produced by a user to model a process(col 46, lines 48-61)"; See also col 3, lines 56-60; col 8, lines 3-12; col 17, lines 36-41); the GPG program receiving information, wherein the information specifies functionality of the new graphical program ("the control information is assigned according to the control functionality associated with the object ...(col 2, lines 63-67)"; " Including a finite state behavior with object definition allows a user to easily designate the specific information of an application program that operates upon object(s) defined by the user (col 4, lines 28-39)"

Kodosky et al. do not explicitly teach that the information does not specify specific objects for the new graphical program.

However, Uczekaj et al. teach that the information not specifying specific objects was known in the art of software development, at the time applicant's invention was made, to allow a user to enter various specific objects ("Including a finite state behavior with object definition allows a user to easily designate the specific information of an

application program that operates upon object(s) defined by the user. The graphical interface tool, which is user friendly, allows a user to enter objects with state information and, based on the entered information, generates application shell code... The generated code is called application shell code because all code is generated except the specific code for any user method names entered in define user method section," col 4, lines 24-39; see also col 3, lines 25-47) such as that disclosed in Uczekaj et al. It would have been obvious for one skilled in the art of computer software development to modify Kodosky's disclosed method to incorporate the teachings of Uczekaj et al. The modification would be obvious because one skilled in the art would be motivated to provide a user to enter various types of information as suggested by Uczekaj et al. in col 4 lines 25-40 and col 10 lines 7-40.

Kodosky et al. further disclose -the GPG program programmatically generating the new graphical program in to response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality ("An executable program is generated in response to the data flow diagram...the executable functions may include user defined functions that have been generated using the method for programming. In this manner, a hierarchy of procedures is implemented, each represented by a data flow diagram (abstract)"; see also col 25, lines 15-37; col 31, lines 41-46).

Regarding claim 2, Kodosky et al. further disclose: said programmatically generating the new graphical program creates the new graphical program without any

user input specifying the new graphical program during said creating ("computer-generated display of a completed block diagram ...This block diagram is the graphical program representing the instrument's operation," col 16, lines 61-67).

Regarding claim 3, Kodosky et al. further disclose: the new graphical program comprises a plurality of interconnected nodes that visually indicate functionality of the new graphical program (abstract; col 8, lines 23-41; col 14, lines19-24, col 16, lines 63-67).

Regarding claim 4, Kodosky et al. further disclose: the new graphical program comprises a block diagram portion comprising a plurality of interconnected nodes (col 14, lines 19-24; abstract) and a user interface portion (col 2, lines 54-62; col 8, lines23-41) wherein said programmatically generating the new graphical program includes generating the block diagram portion and the user interface portion (col 8, lines 23-41).

Regarding claim 5, Kodosky et al. further disclose: said programmatically generating the new graphical program comprises: creating a plurality of graphical program objects in the new graphical program (col 13, lines 63-67; col 14, lines 19-24) and interconnecting the plurality of graphical program objects in the new graphical program (col 14, lines 19-24) wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program (col 13, lines 63-67; col 14, lines 19-24; col 16, lines 61-67; col 17, lines 1-6).

Regarding claim 6, Kodosky et al. further disclose: said programmatically generating the new graphical program comprises: creating one or more user interface objects in the new graphical program (col 2, lines 54-62; col 8, lines23-41), wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program (col 8, lines 23-41).

Regarding claim 7, Kodosky et al. further disclose: the new graphical program is a virtual instrument (col 8, lines 23-25).

Regarding claim 8, Kodosky et al. further disclose: The method of claim 1, wherein the GPG program is a graphical program (col 16, lines 63-67; col 46, lines 56-61).

Regarding claim 9, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies a computational process; wherein the GPG program is operable to generate a new graphical program that implements the specified computational process (col 11, lines 27-43; col 10, lines 31-57).

Regarding claim 10, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies an algorithm; wherein the GPG program is operable to generate a new graphical program that implements the

specified algorithm. Algorithm is a finite set of well-defined rules, procedures, or instructions for the solution of a problem. Kodosky et al. disclose, "It also includes an editor for displaying at least one diagram and for constructing execution instructions. The diagram graphically displays a procedure by which the one or more input variables can produce the one or more output variables (col 3, lines 38-62)." Therefore, accordingly, Kodosky et al. anticipate this claim. See also col 46, lines 40-67; col 47, lines 1-5; col 26, lines 29-32.

Regarding claim 11, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies a state diagram; wherein the GPG program is operable to generate a new graphical program that implements the specified state diagram (col 37, lines 60-67; col 38, lines 1-11).

Regarding claim 12, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies a prototype; wherein the GPG program is operable to generate a new graphical program that implements the specified prototype (col 3, lines 38-62 ;col 46, lines 40-67; col 47, lines 1-5; c01 26, 29-32).

Regarding claim 13, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies a test executive sequence (col 16, lines 50-60); wherein the GPG program is operable to generate a

new graphical program that implements the specified test executive sequence (col 38, lines 41, lines 55; col 16, lines 50-60).

Regarding claim 14, Kodosky et al. further disclose: The method of claim 1, wherein said GPG program receiving information comprises the GPG program receiving user input specifying desired functionality of the new graphical program ("the executable functions may include user defined functions that have been generated using the method for programming (abstract)"; wherein the GPG program is operable to generate a new graphical program that implements the specified desired functionality ("the executable functions may include user defined functions that have been generated using the method for programming. In this manner, a hierarchy of procedures is implemented, each represented by a data flow diagram (abstract)"; see also col 8, lines 23-41).

Regarding claim 15, Kodosky et al. further disclose: The method of claim 14, wherein the GPG program comprises a graphical programming development environment application (col 7, lines 66-67; col 8, lines 1-22).

Regarding claim 16, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program specifies an instrumentation function (col 30, lines 39-54); wherein the GPG program is operable to generate a new

graphical program that implements the specified instrumentation function (col 31, lines 31-46).

Regarding claim 17, Kodosky et al. further disclose: The method of claim 16, wherein the instrumentation function comprises one or more of: a test and measurement function; or an industrial automation function (col 16, lines 50-60; col 17, lines 18-27).

Regarding claim 18, Kodosky et al. further disclose: The method of claim 1, wherein the information received by the GPG program comprises information regarding an existing program having program functionality (col 28, lines 35-50); wherein the GPG program is operable to generate a new graphical program that implements at least a portion of the program functionality of the existing program (col 42, lines 41-49; col 18, lines 5-16; col 14, lines 25-44).

Regarding claim 19, Kodosky et al. further disclose: The method of claim 18, wherein the existing program is a graphical program (col 28, lines 35-50; col 18, lines 5-17; col 39, lines 63-67; col 40, lines 1-3).

Regarding claim 20, Kodosky et al. further disclose: The method of claim 1, wherein the GPG program is operable to generate a plurality of new graphical programs, depending on the received information (col 18, lines 5-16)

Regarding claim 21, Kodosky et al. further disclose: The method of claim 1, wherein the new graphical program generated by the GPG program has program functionality; wherein the GPG program is operable to determine at least a portion of the program functionality independently of the received information (col 30, lines 55-67; col 31, lines 1-13; col 11, lines 19-26; col 32, lines 12-24; col 11, lines 53-60; col 20, lines 38-44).

Regarding claim 22, Kodosky et al. further disclose: The method of claim 1, wherein the GPG program is operable to generate the new graphical program such that the new graphical program implements additional functionality in addition to the functionality specified by the received information (col 22, lines 65-67; col 23, lines 1-2).

Regarding claim 23, Kodosky et al. further disclose: The method of claim 1, wherein the new graphical program comprises graphical program code; wherein the GPG program is operable to receive code generation information specifying how to generate at least a portion of the graphical program code (col 32, lines 25-29; col 33, lines 8-19; col 33, lines 35-45).

Regarding claim 24, Uczekaj et al. further disclose: The method of claim 1, wherein said GPG program programmatically generating the new graphical program comprises the GPG program calling an application programming interface (API) enabling the programmatic generation of a graphical program ("The final files created

are application program interface (API) code files. The API code files provide a level of abstraction between the generated files (CORBA or RPC) and the created class information with any associated class control. The API code files allow state-based control to be directly associated with the object (col 16, lines 36-60)").

Regarding claim 25, Uczekaj et al. further disclose: The method of claim 1, wherein said GPG program programmatically generating the new graphical program comprises the GPG program programmatically requesting a server program to generate the new graphical program (col 3, lines 25-33; .col 8, lines 9-23, lines 65-67; col 9, lines 1-7).

Regarding claim 26, Uczekaj et al. further disclose: The method of claim 25, wherein the server program is an application instance of a graphical programming environment (col 3, lines 25- 33; col 8, lines 9-23; col 5, lines 52-67; col 6, lines 48-62).

Regarding claim 27, Uczekaj et al. disclose: The method of claim 1, wherein the GPG program comprises a client portion and a server portion; wherein the client portion is operable to utilize an application programming interface (API) in order to direct the server program to programmatically generate the new graphical program (col 16, lines 36-60).

Regarding claim 28, Uczekaj et al. further disclose: The method of claim 27, wherein the client portion of the GPG program executes in a first computer system; wherein the server portion of the GPG program executes in a second computer system; wherein the first computer system is connected to the second computer system (col 4, lines 51-64; col 11, lines 19-41; col 16, lines 36-60).

Regarding claim 29, Kodosky et al. further disclose: The method of claim 1, further comprising: executing the new graphical program; wherein the new graphical program is operable to perform the specified functionality during execution (abstract; see the rejection of claim 1 above)

Regarding claim 30, Kodosky et al. further disclose: The method of claim 1, wherein the new graphical program implements only a portion of the specified functionality (col 8, lines 42-54; col 18, lines 30-52; col 21, lines 56-65; col 28, lines 35-50; col 30, lines 7-21).

Regarding claim 31, Kodosky et al. further disclose: The method of claim 1, wherein the new graphical program is a partial program ("The virtual instrument ... includes a front panel ... Which permits interactive use of the virtual instrument ... by a user (col 8, lines 23-41)"; "This capability allows the user to construct a hierarchy of virtual instruments by including previously implemented virtual instruments as "part" in a new instrument (col 28, lines 35-58; See also col 14, lines 45-51; col 17, lines 36-

41)," the method further comprising: adding additional graphical code to the new graphical program, in response to manual user input, in order to complete the new graphical program (col 28, lines 35-58; col 46, lines 40-61).

Regarding claim 32, see the rejection of claim 5 above.

Regarding claim 33, see the rejection of claim 3 above.

Regarding claim 34, Kodosky et al. further disclose: The method of claim 32, wherein the new graphical program includes a block diagram, wherein the at least one graphical program object comprises a programmatic structure placed in the block diagram. Kodosky et al. disclose one of programmatic structures, loop structure ("A sequence structure or a conditional structure contains one or more sub-diagrams and an iterative loop structure or indefinite loop structure contains exactly one diagram. Line 8m indicates that an instrument use node is used to reference another virtual instrument... Line 8u indicates that each object of the node class contains a multiplicity of terminals." See col 14, lines 25-44).

Regarding claim 35, Kodosky et al. further disclose: The method of claim 32, wherein the new graphical program includes a user interface panel, wherein the at least one graphical program object comprises a user interface object placed in the user interface panel (col 18, lines 5-17; col 28, lines 34-50).

Regarding claim 36, Kodosky et al. further disclose: The method of claim 35, wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of:viewing input to the new graphical program; or providing input to the new graphical program (col 40, lines 25-36).

Regarding claim 37, Kodosky et al. further disclose: The method of claim 35, wherein the user interface object is a user interface output object placed in the user interface panel for viewing output of the new graphical program (col 40, lines 25-36; col 8, lines 23-41; col 31, lines 31-46; col 39, lines 41-45).

Regarding claim 38, Kodosky et al. further disclose: The method of claim 35, wherein the new graphical program also includes a block diagram, wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of: viewing input to the block diagram; or providing input to the new graphical program(col 18, lines 5-17; col 28, lines 34-50).

Regarding claim 39, see the rejection of claim 37 above.

Regarding claim 40, Kodosky et al. further disclose: The method of claim 1, wherein said programmatically generating the new graphical program comprises: including a first graphical program object and a second graphical program object in the

new graphical program and connecting the first graphical program object to the second graphical program object (col 27, lines 1-38).

Regarding claim 41, Kodosky et al. further disclose: The method of claim 40, wherein said connecting the first graphical program object to the second graphical program object comprises connecting an input of the first graphical program object to an output of the second graphical program object (col 10, lines 31-57; col 29, lines 17-22; col 31, lines 31-46, lines 55-67; col 46, lines 40-61).

Regarding claim 42, Kodosky et al. further disclose: The method of claim 1, wherein the GPG program is a graphical program (See the rejection of claim 8 above) wherein the GPG program includes at least one object creation node for programmatically creating at least one graphical program object in the new graphical program; wherein said generating the new graphical program comprises including the at least one graphical program object in the new graphical program (See the rejection of claims 6, 32, 34, 38, and 40 above).

Regarding claim 43, Kodosky et al. further disclose: The method of claim 42, wherein the GPG program further includes a property node, the method further comprising: the property node getting or setting a property of the graphical program object in response to said executing the GPG program (col 13, lines 63-67; col 21, lines 26-30; col 24, lines 30-55; col 35, lines 20-30).

Regarding claim 44, Kodosky et al. further disclose: The method of claim 43, wherein the object creation node outputs a reference to the graphical program object; wherein the property node receives the reference as input to the graphical program object; wherein the property node gets or sets a property of the graphical program object specified by the reference to the graphical program object (col 46, lines 40-61; col 16, lines 10-29; col 35, lines 20-30; col 35, lines 65-67; col 36, lines 1-14).

Regarding claim 45, Kodosky et al. further disclose: The method of claim 42, wherein the GPG program further includes an invoke node; the method further comprising: the invoke node invoking a method on the graphical program object in response to said executing the GPG program (col 9, lines 57-64; col 35, lines 44-64; col 36, lines 15-20; col 37, lines 9-24).

Regarding claim 46, Kodosky et al. further disclose: The method of claim 45, wherein the object creation node outputs a reference to the graphical program object; wherein the invoke node receives as input the reference to the graphical program object; wherein the invoke node invokes a method on the graphical program object specified by the reference to the graphical program object (col 46, lines 40-61).

Regarding claim 47, Kodosky et al. further disclose: The method of claim 42, further comprising: configuring the object creation node of the GPG program; wherein said configuring comprises specifying a graphical program object class for the object creation node; wherein the at least one graphical program object included in the new

graphical program is of the graphical program object class (col 17, lines 36-41; col 9, lines 32-44; col 13, lines 63-67; col 29, lines 23-57).

Regarding claim 48, see the rejection of claims 8 and 42.

Regarding claim 49, see the rejection of claim 1 above.

Regarding claim 50, Kodosky et al. further disclose: The method of claim 49, wherein said modifying the existing graphical program comprises adding graphical code to the existing graphical program (col 33, lines 46-58; col 22, lines 65-67; col 41, lines 66-67; col 42, lines 1-4; col 40, lines 25-36; col 42, lines 40-49).

Regarding claim 51, Kodosky et al. further disclose: The method of claim 49, wherein said programmatically modifying the existing graphical program modifies the existing graphical program without any user input specifying the modification to the existing graphical program during said
modifying (col 28, lines 60-65; col 29, lines 1-8). Also, see the rejection of claim 2.

Regarding claim 52, see the rejection of claim 14 above.

Regarding claims 53, 56, and 60, see the rejection of claim 1 above.

Regarding claims 54, 57, and 61, see the rejection of claim 2 above.

Regarding claims 55, 58, and 62, see the rejection of claim 3 above.

Regarding claims 59 and 63, see the rejection of claim 4 above.

Regarding claim 64, see the rejection of claim 5 above.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

7. Claims 1, 49, 53, 56, 60, and 24-28 are rejected under 35 U.S.C. 102(e) as being anticipated by Uczekaj et al. (US 5,920,718).

Regarding claim 1, Uczekaj et al. disclose: A computer-implemented method for programmatically generating a new graphical program ("A display state diagram button ... located in the button bar of graphical interface tool window ... when activated, allows a user to view a state diagram for the class defined. The information entered into the windows associated with the graphical interface tool window ... is stored in a GUI model that includes structures, linked list, etc. A display generator retrieves the control information stored in the GUI model and displays it in a predefined location on the display for presenting a state diagram, i.e., state names are displayed in ovals representing object states (col 10, lines 7-22)") comprising: executing a graphical program generation (GPG) program (col 2, lines 36-50; col 14, lines 6-28); the GPG program receiving information, wherein the information specifies functionality of the

new graphical program, wherein the information does not specify specific objects for the new graphical program ("Including a finite state behavior with object definition allows a user to easily designate the specific information of an application program that operates upon object(s) defined by the user. The graphical interface tool, which is user friendly, allows a user to enter objects with state information and, based on the entered information, generates application shell code (col 4, lines 24-39)"; see also col 3, lines 25-47); the GPG program programmatically generating the new graphical program in to response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality ("After the user has completed entry of the information into object interface section...and object control section ..., the user can double-check the entered data specific to the object control by displaying the state diagram of the defined class....A display state diagram button ... allows a user to view a state diagram for the class defined (col 10, lines 7-22)"; see also col 1, lines 32-48; col 4, lines 23-39).

Regarding claim 24, Uczekaj et al. disclose: The method of claim 1, wherein said GPG program programmatically generating the new graphical program comprises the GPG program calling an application programming interface (API) enabling the programmatic generation of a graphical program ("The final files created are application program interface (API) code files. The API code files provide a level of abstraction between the generated files (CORBA or RPC) and the created class

information with any associated class control. The API code files allow state-based control to be directly associated with the object (col 16, lines 36-60)").

Regarding claim 25, Uczekaj et al. disclose: The method of claim 1, wherein said GPG program programmatically generating the new graphical program comprises the GPG program programmatically requesting a server program to generate the new graphical program (col 3, lines 25-33; .col 8, lines 9-23, lines 65-67; col 9, lines 1-7).

Regarding claim 26, Uczekaj et al. disclose: The method of claim 25, wherein the server program is an application instance of a graphical programming environment (col 3, lines 25- 33; col 8, lines 9-23; col 5, lines 52-67; col 6, lines 48-62).

Regarding claim 27, Uczekaj et al. disclose: The method of claim 1, wherein the GPG program comprises a client portion and a server portion; wherein the client portion is operable to utilize an application programming interface (API) in order to direct the server program to programmatically generate the new graphical program (col 16, lines 36-60).

Regarding claim 28, Uczekaj et al. disclose: The method of claim 27, wherein the client portion of the GPG program executes in a first computer system; wherein the server portion of the GPG program executes in a second computer system; wherein the

first computer system is connected to the second computer system (col 4, lines 51-64; col 11, lines 19-41; col 16, lines 36-60).

Regarding claims 49, 53, 56 and 60, see the rejection of claim 1 above.

8. Claims 1, 49, 53, 56 and 60 are rejected under 35 U.S.C. 102(e) as being anticipated by Morris et al. (US Patent 5,862,372) hereinafter referred to as "Morris."

Per claim 1:

Morris discloses:

-programmatically generating a new graphical program ("Development of a complete application is accomplished by visually arranging, ordering and interconnecting the objects without the necessity of writing any code," abstract)

-executing a graphical program generation (GPG) program; the GPG program receiving information, wherein the information specifies functionality of the new graphical program ("The system of the invention generates the information needed by the run time program," col 3 lines 30-41)

-the information does not specify specific objects for the new graphical program ("utilizing any objects written to a standard specification. ... it is featureless and extensible," col 3 lines 35-60)

- the GPG program programmatically generating the new graphical program in response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality ("The properties are the information or settings which specify to the object how it will perform," col 5 lines 25-

39; "Development of a complete application is accomplished by visually arranging, ordering and interconnecting the objects without the necessity of writing any code," abstract) as claimed.

Per claims 49, 53, 56 and 60, see the rejection of claim 1 above.

Response to Arguments

9. Applicant's arguments with respect to claims 1-23 and 29-64 rejected under 35 U.S.C. 102(b) as being anticipated by Kodosky et al. have been considered but are moot in view of the new ground(s) of rejection.

10. Applicant's arguments filed 3/24/2004 on the rejection under 35 U.S.C. 102 (e) as being anticipated by Uczekaj et al. have been fully considered but they are not persuasive.

Per claim 1:

The Applicant states that Uczekaj et al. (hereinafter referred to as "Uczekaj") do not disclose "the concept of a graphical program." In response to applicant's argument that the reference fails to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., a graphical program where a graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the program (defined in the specification) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from

the specification are not read into the claims. See In re Van Geuns, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). As such, the claims are read with the broadest reasonable interpretation in mind (Note MPEP 2111).

Uczekaj teaches generating a graphical program generated based on a state diagram ("A display generator retrieves the control information stored in the GUI model and displays it in a predefined location on the display for presenting a state diagram... Once the user is fully satisfied with the information entered into graphical interface tool window... the user activates a generate code button 286 that causes application shell code based on the information entered to be automatically generated. The generated code is called application shell code because all code is generated except the specific code for any user method names entered in define user method section (col 10 lines 23-40; col 3, lines 33-49; col 4, lines 28-39). The examiner interprets a graphical program as a program of or relating to written or visual representation. Therefore, the application shell code generated by activating a generate code button 286 within the graphical interface tool window is interpreted as a graphical program. Accordingly, in view of the broadest reasonable interpretation, Uczekaj discloses programmatically generating a graphical program based on a state diagram as claimed. Therefore, the rejection of claim 1 is considered proper and maintained.

Per claims 53, 56 and 60:

The applicant states that Uczekaj does not disclose the limitations of claims 49, 53, 56 and 60 for the reasons set forth in connection with claim 1. As shown above, the

rejection of claim 1 by Uczekaj was maintained, and accordingly, the rejections of claims 49, 53, 56 and 60 are also maintained.

Per claims 24-28:

The applicant states that claims 24-28 are allowable as being dependent on allowable base claim. As has been shown above, the rejections of the independent claim 1by Uczekaj are proper, the argument that claims 24-28 are allowable as being dependent on an allowable base claim is considered moot.

Accordingly, the rejections of claims 24-28 are proper and maintained.

Conclusion

11. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of

the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Insun Kang whose telephone number is 703-305-6465. The examiner can normally be reached on M-F 8:30-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on 703-305-9662. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

IK
5/14/2004

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100